Individual Assignment 5

Please download the **Running App Data, Training & Validation.csv** and **Running App Data, Test.**csv datasets from Canvas. Save the files in a known location.

These both contain data about a set of users who use a popular running app. Both datasets have the following column definitions:

runner_id	Unique numerical identifier for runner
africa	1 if runner located in Africa, 0 if not
south_america	1 if runner located in South America, 0 if not
asia	1 if runner located in Asia, 0 if not
likes_per_day	Likes per day received on app from other users
is_male	1 if runner is male, 0 if not
marathons	# marathons run in life
ultramarathoner	1 if runner is an ultramarathoner, 0 if not
age	Runner age
distance_per_day	Average running distance logged per day (KM)
premium_member	1 if runner paid for premium membership, 0 if not

The company that owns the running app wants to be able to classify the app users who pay for premium memberships. You will run a naïve Bayes classification analysis in R to do this.

Open RStudio and create an R Markdown file. Save the .rmd file, ensuring that you save it in a file location that you can easily find again.

Each of the following items corresponds to a code chunk, or text to record in the R Markdown file. Please make sure to label each item using the numbers provided in this document.

Make sure to run each code chunk after you write it to ensure that there are no errors. Be careful about typos!

If you have not installed the naivebayes, caret, dplyr, and pROC libraries yet, do so according to the instructions given in class before beginning this portion of the assignment.

Item 1, Loading Packages

[Code Chunk]: Load the naivebayes, caret, dplyr, and pROC libraries.

Item 2, Importing Data

[Code Chunk]: Import the Running App Data, Training & Validation.csv file into an R data frame called “appdata_tv”.

[Code Chunk]: Import the Running App Data, Test.csv file into an R data frame called “appdata_test”.

Individual Assignment 5

Item 3, Preparing Dataset for Classification

[Code Chunk]: For both appdata_tv and appdata_test, set the outcome variable (premium_member) as a categorical variable so that classification can be run.

Item 4, Assessing Outcome Variable Balance in Training & Validation

[Code Chunk]: Run a line of code to assess the balance of the outcome variable (premium_member) in the appdata_tv data frame.

[Text]: In the training & validation data, how many observations are there for premium members? How many observations are there for non-premium members? Is this a balanced or imbalanced dataset?

Item 5, Setting Random Seed

Code Chunk]: Set the random seed to 1234.

Item 6a, Oversampling the Training & Validation Data

[Code Chunk]: Perform oversampling on the appdata_tv data frame. Make sure that the data stratum containing only non-premium members is reduced to 1,450 rows. The other stratum should contain all of the premium members. Combine the reduced non-premium stratum with the premium stratum to produce one oversampled data frame.

Item 6b, Oversampling Check

[Code Chunk]: Run code to assess how many rows are present in the oversampled training & validation data frame.

[Text]: How rows are present in the oversampled training & validation data frame?

Item 7, Data Partitioning

[Code Chunk]: Set the random seed to 1234 again.

[Code Chunk]: Partition the oversampled data into training and validation. 80% of the data should go into training, the rest should go into validation.

Item 8, Training Naïve Bayes Model

[Code Chunk]: Train a naïve Bayes model on the training data partition. The outcome variable is whether a runner is a premium member or not. Use all available predictors except for runner_id.

Item 9, Producing Probability Predictions on Validation & Test Data

[Code Chunk]: Use the naïve Bayes model to produce probability predictions on the validation data as well as the test data.

(Do this using the method shown in class, so that these predictions are appropriately saved as a new column in each data frame.)

Item 10, Obtaining Test Data AUC

[Code Chunk]: Obtain the area under the curve (AUC) for the predictions made on the test data.

[Text]: What is the test AUC? Is this a good AUC?

Item 11, Selecting Probability Threshold Using the Test ROC Curve

[Code Chunk]: Extract the records from the ROC curve data wherein the sensitivity is greater than or equal to 0.80 and the precision is greater than or equal to 0.23. Display these records.

[Text]: What probability threshold should be set to ensure that the sensitivity is greater than or equal to 0.80 and the precision is greater than or equal to 0.23? Choose a probability threshold that maximizes precision under these constraints.

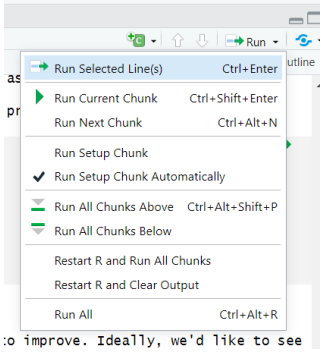
OPTIONAL EXTRA CREDIT PORTION (1 POINT):

Classify all runners in the test data as premium members or not using the probability threshold identified in Item 11.

Based on these classifications, generate and display the confusion matrix and associated performance metrics for the test data predictions.

(EXTRA CREDIT QUESTIONS):
What is the accuracy on the test data?
What is the precision on the test data?
Interpret both metrics.

Click "Run All" to run all of your code chunks and check the output. Make sure that any text answers you have match this output. This is to ensure that your text answers match the output produced by R, factoring in randomization!



Save your file, then knit and export your R Markdown file as an HTML file. Upload the HTML file to Canvas to complete the assignment.

If you are unable to knit your .rmd file due to errors, make sure that you go back and test your code chunks individually. If you are ultimately unable to figure out how to solve these errors, save the .rmd file and upload that instead of the HTML file for partial credit.

If you are unable to find your exported HTML file, consult Canvas for instructions.